Post-Mortem

Assignment 09

December 8, 2017

We normally publish the post-mortem for an assignment after it has been marked and released. Here is a list of common errors provided by the graders for Assignment 9.

General

- Many students did not make use of parametric types, or used them incorrectly in contracts throughout the assignment. Parametric types should be used when the actual type itself can vary, but there is a relationship in the types of values that are consumed and/or produced by a function.

- Many students had issues in formatting their contracts for functions with an argument that can take on more than one type or value. For example, a list that contains only 0’s and 1’s should be written as \((\text{listof } (\text{anyof } 0 1))\) in the contract exactly (if this is not communicated in the requirements).

- Some students made their code unnecessarily complex by wrapping the first element of the produced list in another list, and using \text{append} to combine the wrapped list with the recursive call on the rest. Instead, use \text{cons} to combine the first element of the produced list with the recursive call.

Question 1

- In part (a), many students who did not use the provided hints had an overly complex solution. However, marks were not deducted if the hints were not used.

- In part (b), many students were missing at least one of the requirements for \text{next-row}. It was necessary to specify that the rule number must be less than or equal to 255, that the consumed list must be non-empty, and that each element in the consumed list must either be 0 or 1 (if this wasn’t already specified in the contract).

- In part (c), many students had an incorrect contract. The most common incorrect contracts were “iterate: \((X \rightarrow Y) \ (\text{listof } X) \ \text{Nat} \rightarrow (\text{listof } Y)\), and “iterate: \((\text{Any} \rightarrow \text{Any}) \ (\text{listof } \text{Any}) \ \text{Nat} \rightarrow (\text{listof } \text{Any})\). It is important to realize that the first argument consumed by \text{iterate} should be a function that consumes and produces the same type.

- In part (c), some students made their base case when \(n\) is equal to 1, and thus did not account for the case where \(n\) is zero.

Question 2

- In part (a), some students incorrectly indicated in their contract that \text{map2d} consumes and produces a list of the same type.
• In part (b), almost all students missed the requirement that the consumed \((\text{listof (listof Nat)})\) must be a non-empty list of non-empty lists, where each sub-list is of the same length. Although this was not explicitly stated in the question, \texttt{construct-puzzle} must produce a \texttt{State}, and a \texttt{State} contains a \texttt{Grid} that must satisfy these requirements.

• Many students did not complete parts (e) or (f). Some students hard-coded part (e) to always produce empty, and part (f) to always produce false, but doing so would not receive any correctness marks.

• Many students who completed part (e) did not include design recipe components for all of their helper functions, whether they were defined at the top level, or inside a \texttt{local}.

• In part (f), some students immediately called \texttt{state-rects} after calling the \texttt{search} function with its arguments. However, if \texttt{search} produced false, attempting to select the rectangles would produce an error.

**Ongoing Errors**

The following is a list of common errors from previous assignments that were still repeated for assignment 9.

• Many students who included multiple helper functions inside a \texttt{local} were still missing separators between their function blocks. Not only are separators necessary for functions defined at the top level, but they are also necessary for locally defined functions.

• Some students were still missing design recipe components for locally defined helper functions. Remember that purposes and contracts are required for any functions defined inside a \texttt{local} as well.